Last Time

Multidimensional arrays

ArrayList

HashMap

Iterators

For-each loop

Analysis of Data Structures

Performance trade-offs

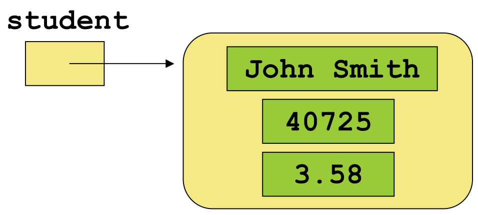
How to build data structures—with references

Object References

Recall that an *object reference* is a variable that stores the address of an object

A reference also can be called a *pointer*

References often are depicted graphically:



References as Links

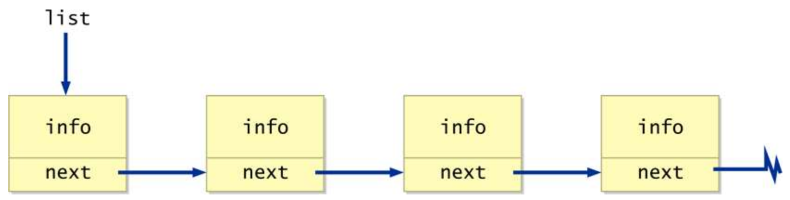
Object references can be used to create *links* between objects

Suppose a class contains a reference to another object of the same class:

```
class Node
{
   int info;
   Node next;
}
```

References as Links

References can be used to create a variety of linked structures, such as a *linked list*:



Intermediate Nodes

The objects being stored should not be concerned with the details of the data structure in which they may be stored

For example, the `Student` class should not have to store a link to the next `Student` object in the list

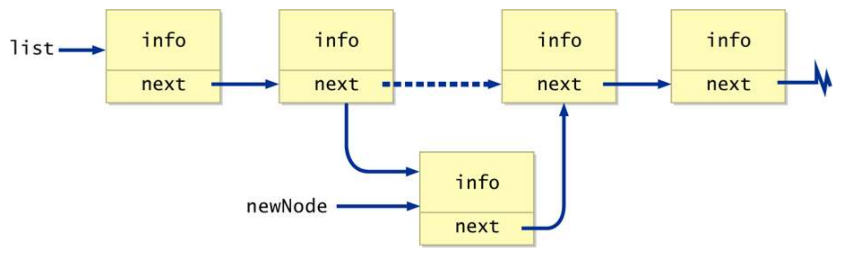
Instead, use a separate node class with two parts:

- a reference to an independent object
- a link to the next node in the list

The internal representation becomes a linked list of nodes

Inserting a Node

A node can be inserted into the middle of a linked list with a few reference changes:



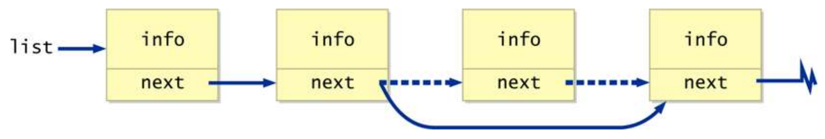
Quick Check

Write code that inserts `newNode` after the node pointed to by `current`.

```
newNode.next = current.next;
current.next = newNode;
```

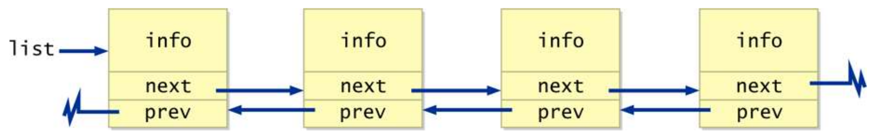
Deleting a Node

Likewise, a node can be removed from a linked list by changing the `next` pointer of the preceding node:



Other Dynamic Representations

It may be convenient to implement a list as a *doubly linked list*, with `next` and `previous` references:



Magazine Collection

Let’s explore an example of a collection of `Magazine` objects, managed by the `MagazineList` class, which has an private inner class called `MagazineNode`

```java
//********************************************************************
//  Magazine.java       Author: Lewis/Loftus
//
//  Represents a single magazine.
//********************************************************************

public class Magazine
{
   private String title;

   //-----------------------------------------------------------------
   //  Sets up the new magazine with its title.
   //-----------------------------------------------------------------
   public Magazine(String newTitle)
   {
      title = newTitle;
   }

   //-----------------------------------------------------------------
   //  Returns this magazine as a string.
   //-----------------------------------------------------------------
   public String toString()
   {
      return title;
   }
}
```

continue

```
//*******************************************************************
//  A class that represents a node in the magazine list.
//  The public variables are accessed by the MagazineList class.
//*******************************************************************
private class MagazineNode
{
   public Magazine magazine;
   public MagazineNode next;

   //--------------------------------------------------------------
   //  Sets up the node
   //--------------------------------------------------------------
   public MagazineNode(Magazine mag)
   {
      magazine = mag;
      next = null;
   }
}
```

```
//********************************************************************
//  MagazineList.java       Author: Lewis/Loftus
//
//  Represents a collection of magazines.
//********************************************************************

public class MagazineList
{
   private MagazineNode list;

   //-----------------------------------------------------------------
   //  Sets up an initially empty list of magazines.
   //-----------------------------------------------------------------
   public MagazineList()
   {
      list = null;
   }
```

continue

continue

```
   //-----------------------------------------------------------------
   //  Creates a new MagazineNode object and adds it to the end of
   //  the linked list.
   //-----------------------------------------------------------------
   public void add(Magazine mag)
   {
      MagazineNode node = new MagazineNode(mag);

      if (list == null)
         list = node;
      else
      {
         MagazineNode current = list;
         while (current.next != null)
            current = current.next;
         current.next = node;
      }
   }
```

continue

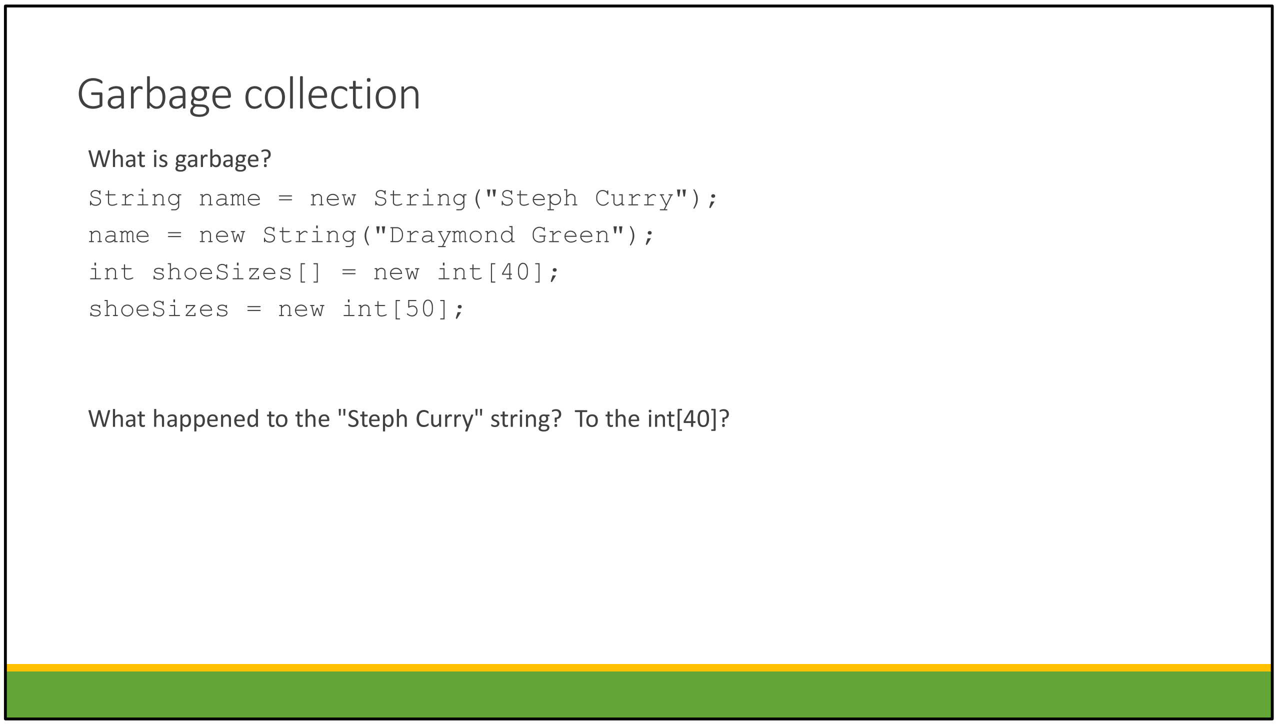
continue

```
   //-----------------------------------------------------------------
   //  Returns this list of magazines as a string.
   //-----------------------------------------------------------------
   public String toString()
   {
      String result = "";

      MagazineNode current = list;

      while (current != null)
      {
         result += current.magazine + "\n";
         current = current.next;
      }

      return result;
   }
} // end class MagazineList
```

```
//***************                                   *****************
//  MagazineRack.
//
//  Driver to exe
//***************                                   *****************

public class Maga
{
   //-----------------------------------------------------------------
   //  Creates a MagazineList object, adds several magazines to the
   //  list, then prints it.
   //-----------------------------------------------------------------
   public static void main(String[] args)
   {
      MagazineList rack = new MagazineList();

      rack.add(new Magazine("Time"));
      rack.add(new Magazine("Woodworking Today"));
      rack.add(new Magazine("Communications of the ACM"));
      rack.add(new Magazine("House and Garden"));
      rack.add(new Magazine("GQ"));

      System.out.println(rack);
   }
}
```

Output

```
Time
Woodworking Today
Communications of the ACM
House and Garden
GQ
```



Garbage collection

What is garbage?

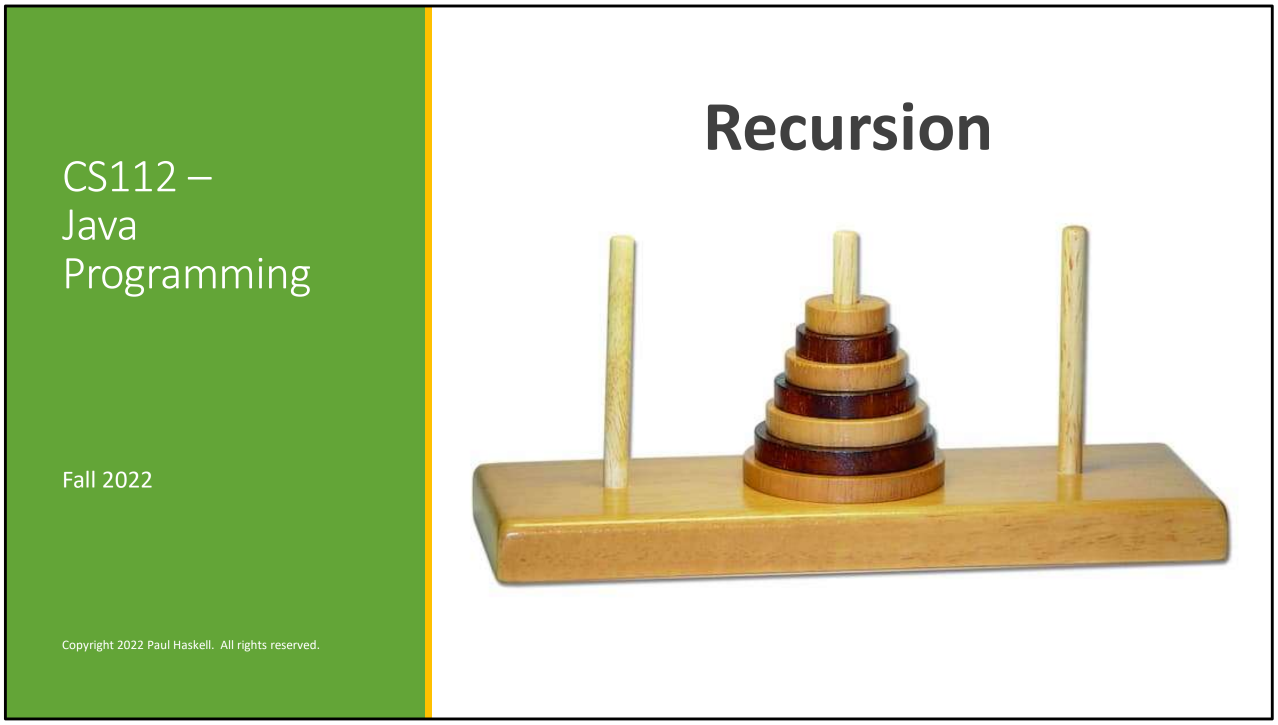
```
String name = new String("Steph Curry");
name = new String("Draymond Green");
int shoeSizes[] = new int[40];
shoeSizes = new int[50];
```

What happened to the "Steph Curry" string? To the int[40]?

Objects that don't have any references referring to them anymore are “garbage”. They cannot be used by the program and they just consume memory.
Java objects (strings, arrays, user classes, etc) are tracked by the Java runtime.
Occasionally, the JRE goes through its list of garbage and "frees" it, making the memory available for other uses. In C, C++, and older languages, the programmer must do this herself, leading to a never-ending series of program crashes and bugs.
So now you know.

Interlude

Input file redirection review

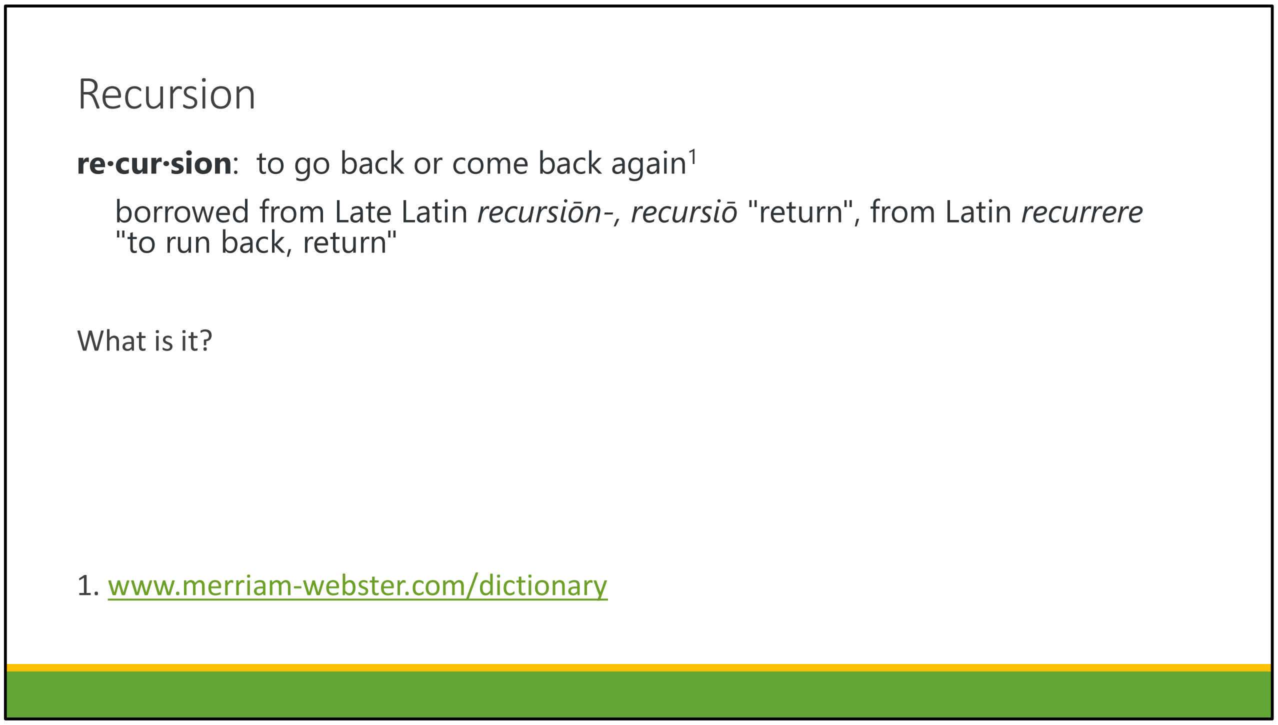


CS112 – Java Programming

Fall 2022

Recursion

Advanced data structures
What's a data structure? A structure for storing, retrieving, and manipulating data, with desired properties
Data, list, stack, tree, dictionary, etc



Recursion

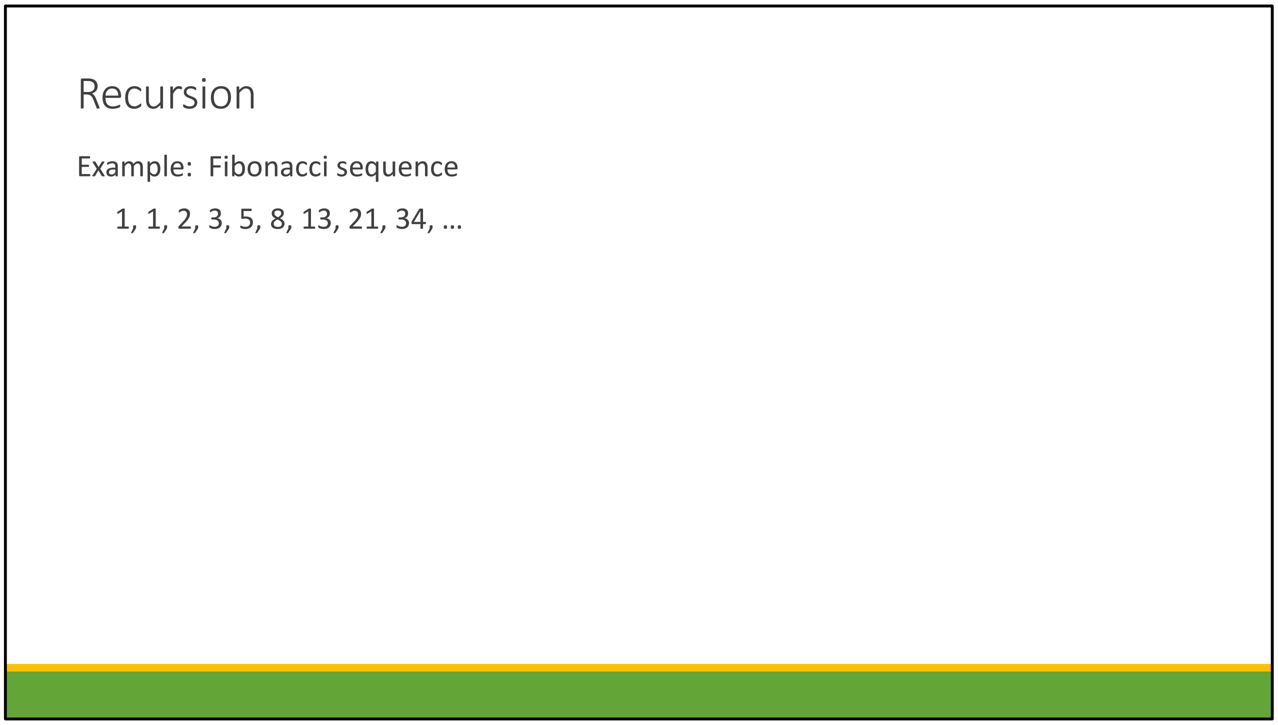
re·cur·sion: to go back or come back again[1]

borrowed from Late Latin *recursiōn-, recursiō* "return", from Latin *recurrere* "to run back, return"

What is it?

1. www.merriam-webster.com/dictionary

Solving a problem in terms of a slightly smaller version of the same problem
Example: factorial
"End condition"



Recursion

Example: Fibonacci sequence

1, 1, 2, 3, 5, 8, 13, 21, 34, …

Recursion is a way to think about solving some problems. If it can be used, solution can be elegant. But obviously a bit tricky to think about. You will bet getting some experience thinking about solving problems in recursive ways during the rest of the semester.
DON'T OVERUSE! Is useful "mind map" and is definitely used, but is tricky enough that it should be used when needed, not just for fun.
Like inheritance